Program Assignment 3: Intro to Deep Learning

Deep Artificial Neural Networks on MNIST dataset

Due on Monday, Mar. 23th, at the beginning of class

Professor Qiang Ji

Keyi Liu(Master Student)

March 21, 2017

1 Assignment Introduction

In this programming assignment, I have implemented techniques to learn a deep(two hidden layers) neural network(NN) for image digit classification on the MNIST dataset. The NN will take an image of a hand-written numerical digit as the input and classify it into one of 10 classes corresponding to digits 0 to 9 respectively. Given the training data, I followed the equations in the lecture notes to train the NN using the back propagation method and then evaluate its performance on the given testing data. The NN has one input layer(784 nodes), two hidden layers with 100 nodes for each layer, and one output layer with 10 nodes, corresponding to the 10 digit classes respectively.

For training, the parameters are learned by minimizing the squared loss for the output nodes. There are two main process during the training, perform the forward propagation first, and then goes the backward propagation to update the parameters.

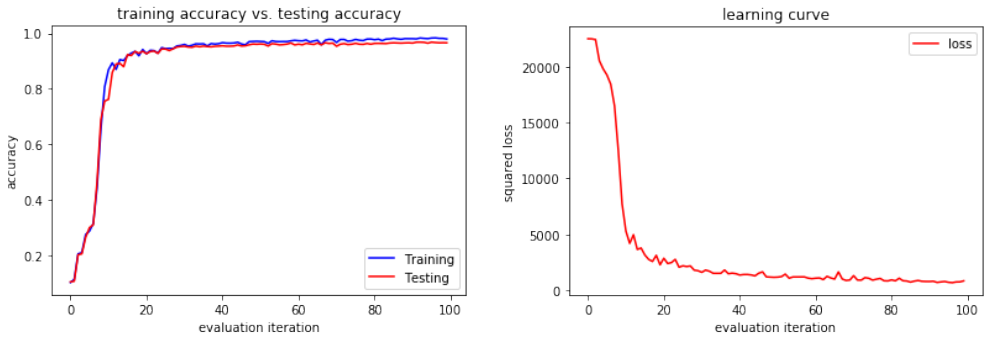
2 Specification and results

Specifically, given the training data $\mathcal{D} = \{x[m], y[m]\}$, $m = 1, 2, \cdots, M$, where $x[m]$ is a grayscale image of $28 \times 28 (784 \times 1$ vector) and $y[m]$ is output vector that follows the 1 of K encoding, with k^{th} element of being 1 and the rest being 0. Use $ReLU$ activation function for the hidden nodes and the $softmax$(multi-class sigmoid) function for the output.

For a deep neural network, tuning the hyper parameter can be really tricky. I pick the learning rate as 0.02 at the beginning, and decay it by 0.5 every epoch, the regularization factor is 0.000001, within each epoch, I also decay it by a factor related to the number of iteration and another constant decay factor.

(1) I use the stochastic gradient method, that is, randomly choosing a mini-batch of 50 data from $\mathcal{D}$. For each point in the mini-batch, perform forward propagation and back propagation to compute the gradients of weight matrix and weight bias vector for each layer. Then update the weights and bias using the average gradient of the gradients computed from all the 50 samples.

(2) Using the given testing dataset T, evaluate the performance of the trained NN by computing the classification error for each digit as well as the average classification errors for all digits. The classification error for each digit is computed as the ratio of incorrect classification to the total number images for that digit. Plot the average training classification error, average testing classification error, and value of the loss function after each parameters update. Here I show the plot of the training and testing accuracy, and after 200 evaluation, the training accuracy is 0.9830, and the testing accuracy is 0.9682.



(a) learning vs. testing

(b) learning curve

Figure 1: Experimental results

(3) The test accuracy with respect to each class, and the overall accuracy are listed in the following table,

Digits	0	1	2	3	4	5	6	7	8	9	overall
Accuracy	0.9940	0.9813	0.9678	0.9463	0.9566	0.9497	9876	9639	9650	0.9644	0.9682

Table 1: testing accuracy of each digit